Generating Lefkovitch models

2025-03-03

Introduction

Lefkovitch matrix population models (MPMs) were introduced by Leonard Lefkovitch in his 1965 paper, "*The Study of Population Growth in Organisms Grouped by Stages*", published in *Biometrics.*. This paper extended the concept of Leslie MPMs, which are structured by age, to stage-structured populations, providing a framework that has been widely used in ecology, evolution and conservation studies.

In a Lefkovitch MPM, the square matrix is used to model population growth across discrete projection intervals, typically representing years. Each matrix element represents either a transition probability between different stages or the reproductive output of a stage across the projection interval. The MPM can be divided into submatrices: one for survival/growth (the $\mathbf{U}$ matrix), one for sexual reproduction (the $\mathbf{F}$ matrix) and one for asexual reproduction (the $\mathbf{C}$ matrix), where $\mathbf{A} = \mathbf{U} + \mathbf{F} + \mathbf{C}$. Occasionally, these reproduction matrices are lumped together as a reproduction matrix, $\mathbf{R}$ (i.e. $\mathbf{R} = \mathbf{F} + \mathbf{C}$). Reproduction is often termed fecundity in the population biology literature.

The elements of the $\mathbf{U}$ matrix represent survival or growth from stage-to-stage between time steps. Therefore the column sums of the $\mathbf{U}$ submatrix cannot exceed 1. The reproductive output elements in the $\mathbf{F}$ and $\mathbf{C}$ (or $\mathbf{R}$) submatrices do not have an upper bound and indicate the number of new individuals each stage can produce in each time interval.

Zero entries in the matrices indicate that those transitions do not occur.

To project population size and structure over time, the MPM is multiplied by a vector representing the current population structure (number of individuals in each stage). This results in a new vector that shows the predicted population structure for the next time step. This process can be repeated to project population dynamics over multiple time steps.

Lefkovitch models are useful for studying population dynamics under different scenarios, such as changes in survival or reproductive rates, or different management strategies. They have broad applications in both theoretical and applied ecology.

Aims

The purpose of this vignette is to illustrate how to simulate stage-based (Lefkovitch) MPMs based on defined life history archetypes. There are several reasons why one would want to do this, including, but not limited to:

- Exploring how life history or life cycle structure influences population dynamics.
- Generating MPMs with defined life cycle properties for teaching purposes.

In the following sections, this vignette will:

1) Explain how life cycles can be categorised into Archetypes.
2) Show how to generate a random Lefkovitch MPM based on an archetype.
3) Show how to rapidly produce sets of many random MPMs.
4) Show how to constrain the MPMs by matrix properties.

Preparation

Before beginning, users will need to load the required packages.

```
library(mpmsim)
library(dplyr)
library(Rage)
library(ggplot2)
library(Rcompadre)
```

1. Life cycle archetypes and generating an MPM

In stage-based (Lefkovitch) matrix population models (MPMs), different life cycle types can be represented by various structural forms of the matrices. These life cycle types can be captured using different life history archetypes, which define the transitions between stages and the survival probabilities in the population.

The life history archetypes, based on Takada et al. (2018), are as follows:

- Archetype 1: All elements are positive, meaning transitions from/to any stage are possible. This model represents a life history where individuals can progress and retrogress rapidly.
- Archetype 2: Similar to Archetype 1, but with the survival probability increasing monotonically as individuals advance to later stages. This model also allows for rapid progression and retrogression but with more realistic stage-specific survival probabilities.
- Archetype 3: Positive non-zero elements for survival are only allowed on the diagonal and lower sub-diagonal of the matrix. This model represents a life cycle where retrogression is not allowed, and progression can only happen to the immediately larger/more developed stage (slow progression, e.g., trees).
- Archetype 4: Similar to Archetype 3 but with the additional assumption that stage-specific survival increases as individuals increase in size/developmental stage.

In all these archetypes, fecundity is placed in the top row of the matrix. In Takada et al.'s paper, fecundity was always placed in the top right of the MPM, meaning that only the "last" stage of the life cycle reproduced. This approach can be relaxed to allow reproduction from any stage.

2. Generate a random Lefkovitch MPM based on an archetype

In `mpmsim` the function `rand_lefko_mpm` can be used to generate a random MPM that conforms to one of the above four life cycle archetypes. The function allows for the generation of random MPMs based on these archetypes, with survival and growth (the **U** matrix) based on draws from a Dirichlet distribution to ensure biological plausibility. The Dirichlet distribution is used to draw survival and growth values because it ensures that the sum of the probabilities for each stage is equal to 1, which is necessary for biologically realistic models. The function allows users to specify a wide range of reproductive output scenarios , offering flexibility in how fecundity is modelled across stages.

The function is straightforward. In the following example, I create a three-stage MPM based on Archetype 2. I set fecundity, arbitrarily, to be 5. By default, if only a single value is given, this is placed in the top-right hand corner of the **F** matrix. Also, by default, all fecundity is assumed to be sexual.

```
rand_lefko_mpm(n_stages = 3, fecundity = 5, archetype = 2)
#> $mat_A
#>              [,1]       [,2]        [,3]
#> [1,] 0.005877166 0.10766181 5.72412770
```

```
#> [2,] 0.509913175 0.03602856 0.06736799
#> [3,] 0.287840153 0.74138349 0.19955576
#>
#> $mat_U
#>             [,1]       [,2]       [,3]
#> [1,] 0.005877166 0.10766181 0.72412770
#> [2,] 0.509913175 0.03602856 0.06736799
#> [3,] 0.287840153 0.74138349 0.19955576
#>
#> $mat_F
#>      [,1] [,2] [,3]
#> [1,]    0    0    5
#> [2,]    0    0    0
#> [3,]    0    0    0
```

To introduce variability in fecundity, users can provide reproductive output as a list of two matrices, with numeric values of the same dimensions as n_stages, representing the lower and upper limits of mean fecundity for the entire matrix model. Reproductive output values are then drawn from a uniform distribution between the two values. Users should use 0 for both lower and upper limits in cases with no fecundity.

The following code provides an example:

```
lower_reprod <- matrix(c(
  0, 0, 0,
  0, 0, 0,
  0, 0, 0
), nrow = 3, ncol = 3, byrow = TRUE)
upper_reprod <- matrix(c(
  0, 4, 20,
  0, 0, 0,
  0, 0, 0
), nrow = 3, ncol = 3, byrow = TRUE)

rand_lefko_mpm(
  n_stages = 3, fecundity = list(lower_reprod, upper_reprod),
  archetype = 2
)
#> $mat_A
#>            [,1]       [,2]        [,3]
#> [1,] 0.0873477 2.6402006 15.2043937
#> [2,] 0.4082913 0.1851252  0.5516964
#> [3,] 0.3924441 0.1736190  0.1107596
#>
#> $mat_U
#>            [,1]       [,2]       [,3]
#> [1,] 0.0873477 0.5799473 0.3249008
#> [2,] 0.4082913 0.1851252 0.5516964
#> [3,] 0.3924441 0.1736190 0.1107596
#>
#> $mat_F
#>      [,1]       [,2]      [,3]
#> [1,]    0 2.060253 14.87949
#> [2,]    0 0.000000  0.00000
#> [3,]    0 0.000000  0.00000
```

3. Generate sets of Lefkovitch matrices

It is sometimes desirable to create large numbers of MPMs with particular properties in order to test hypotheses. For stage-based (Lefkovitch) MPMs, this can be implemented using the `rand_lefko_set` function. This function acts as a wrapper for the previously described function and generates a set of Lefkovitch MPMs based on a defined life cycle archetype and specified reproductive output. For example, users may wish to generate MPMs for different life history archetypes to explore how life cycle structure may influence population dynamics. By specifying the number of stages, fecundity values, and archetypes, users can produce MPMs that are tailored to their specific research needs. This capability is useful for exploring the effects of life history traits on population dynamics, testing ecological and evolutionary hypotheses, and for teaching purposes.

The function returns either a `list` or a `CompadreDB` object depending on the `output` argument. If the output is set as a `CompadreDB` object, the archetype of the MPM is included as a column of metadata.

The following code shows how users can generate 100 matrices in a `CompadreDB` object.

```
myMatrices <- rand_lefko_set(
  n = 100, n_stages = 3, fecundity = 12,
  archetype = 4, output = "Type1"
)
```

After producing the output as a `CompadreDB` object, the matrices can be accessed using functions from the `RCompadre` R package. For example, to get the **A** matrix, or the **U**/**F** submatrices users can use the `matA`, `matU` or `matF` functions. The following code illustrates how to rapidly calculate population growth rate for all of the matrices.

```
# Obtain the matrices
x <- matA(myMatrices)

# Calculate lambda for each matrix
lambdaVals <- sapply(x, popdemo::eigs, what = "lambda")
summary(lambdaVals)
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>  0.7378  1.1395  1.4965  1.4653  1.7106  2.1926
```

Users can examine the vignettes for the `Rcompadre` and `Rage` packages for additional insight into other potential operations with the `compadreDB` object.

4. Constraining the output matrices

Critically, users can impose constraints on the "acceptable" properties of these randomly generated MPMs. For example, in some analyses, it may be desirable to generate MPMs where the population growth rate is constrained to values near 1.

This is handled by the `constraint` argument, which takes a data frame specifying the criteria for acceptable MPMs. The data frame must have four columns: `fun`, `arg`, `lower`, and `upper`. The `fun` column contains the name of the function that calculates the metric to be constrained (e.g., `eigs`, from the `popdemo` package). The `arg` column specifies any additional argument that the function requires (e.g., `what = "lambda"` for the `eigs` function), using `NA` if no additional argument is needed. The `lower` and `upper` columns set the bounds of the acceptable range for the metric.

Here's an example of how to use the constraint argument to ensure that the generated MPMs have a population growth rate (lambda) between 0.9 and 1.1.

```r
library(popdemo)

constrain_df <- data.frame(
  fun = "eigs", arg = "lambda", lower = 0.9, upper = 1.1
)

myMatrices <- rand_lefko_set(
  n = 100, n_stages = 3, fecundity = 12, constraint = constrain_df,
  archetype = 4, output = "Type1"
)
```

We can check that it has worked by examining the matrices.

```r
# Obtain the matrices
x <- matA(myMatrices)

# Calculate lambda for each matrix
lambdaVals <- sapply(x, popdemo::eigs, what = "lambda")
summary(lambdaVals)
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>  0.9019  0.9612  1.0148  1.0115  1.0635  1.0984
```

Conclusion

This vignette has provided a comprehensive guide to generating Lefkovitch matrix population models (MPMs) based on life history archetypes. By using the `rand_lefko_mpm` and `rand_lefko_set` functions, users can create individual MPMs or large sets of MPMs tailored to specific research needs. The ability to impose constraints on these models allows for precise control over their properties, ensuring that generated MPMs meet defined criteria, such as specific population growth rates.

The flexibility and power of these functions facilitate the exploration of population dynamics under various scenarios, aiding in hypothesis testing in studies of population biology and life history theory. Additionally, tight integration with the `RCompadre` package facilitates the use of generated models, enhancing their utility in both theoretical and applied ecological research.